Amendment to Claims

1-14 (canceled).

15 (currently amended). The processor Claim 73 [[14]] wherein the processor executes the ~~first~~ associated instruction to completion when the resource ~~FIFO~~ becomes available.

16-21 (canceled).

22 (currently amended). The processor of Claim [[19]] 71 wherein the ~~second~~ circuitry is to schedule a task or tasks for execution on each instruction executed by the processor such that whenever the processor is to execute any instruction, the ~~second~~ circuitry is to perform the task scheduling to schedule a task that will execute the instruction.

23 (canceled).

24 (currently amended). The method of Claim [[23]] 78 wherein the ~~first~~ associated instruction is re-executed when the resource ~~FIFO~~ becomes available.

25-28 (canceled).

29 (currently amended). The method of Claim [[26]] 76 wherein scheduling a task or tasks for execution is performed on each instruction executed by any one of the tasks such that whenever an instruction is to be executed, the task scheduling is performed to schedule a task that will execute the instruction.

30 (currently amended). The processor of Claim [[14]] 71 wherein:

the processor comprises an instruction execution unit for executing instructions stored in a memory and fetched from the memory; and

on obtaining of any one of the first and second indications, the processor is operable to:

(a) ~~the processor suspends~~ suspend the ~~first~~ associated instruction after the ~~first~~ associated instruction has been fetched from the memory; and

(b) ~~the processor again fetches~~ re-fetch the ~~first~~ associated instruction from the memory to re-execute the ~~first~~ associated instruction.

31 (currently amended). The processor of Claim [[14]] 71 wherein the ~~first~~ associated instruction is suspended after being decoded by the processor, and the ~~first~~ associated instruction is decoded again when the ~~first~~ associated instruction is being re-executed by the processor.

32-37 (canceled).

38 (currently amended). The processor of Claim [[37]] 71 wherein suspension of the ~~first~~ associated task and scheduling of ~~the other~~ another task ~~instead~~ in lieu of the ~~first~~ associated task does not involve instruction execution by the processor.

39-41 (canceled).

42 (currently amended). The processor of Claim [[17]] 71 wherein the processor comprises an instruction execution pipeline operable to concurrently execute instructions from different tasks, wherein on obtaining of any one of the first and second indications the processor is operable to suspend the ~~first~~ associated instruction after the pipeline starts execution of one or more other instructions for one or more tasks other than the ~~first~~ associated task but before the pipeline starts execution of any instruction following the ~~first~~ associated instruction for the ~~first~~ associated task.

43 (currently amended). The processor of Claim 42 wherein the pipeline comprises an instruction decode stage, and the ~~first~~ associated instruction is suspended after being processed by the decode stage.

44 (currently amended). The processor of Claim 42 wherein the ~~first~~ associated instruction is suspended in the pipeline's read stage in which instruction operands are read from storage and presented to instruction execution logic.

45 (currently amended). The processor of Claim [[17]] 73 wherein the resource is a FIFO [[is]] shared by the ~~first~~ associated task and at least one other task.

46 (currently amended). The processor of Claim [[18]] 73 wherein the resource is a FIFO.

47 (currently amended). The processor of Claim 46 wherein the FIFO is a request FIFO for storing requests to process network data, and for at least one first indication the ~~first~~ associated instruction is an instruction to read the request FIFO.

48 (currently amended). The processor of Claim 46 wherein the FIFO is a command FIFO for storing commands for processing of network data, and for at least one first indication the ~~first~~ associated instruction is an instruction to write one or more commands to the command FIFO.

49 (currently amended). The processor of Claim 46 wherein the FIFO is a status FIFO for storing status information on reception of network data over a network, and for at least one first indication the ~~first~~ associated instruction is an instruction to read the status FIFO.

50 (currently amended). The method of Claim [[23]] 76 wherein ~~said signal~~ each of the first and second indications is received after the ~~first~~ associated instruction has been fetched from a memory for execution; and

~~the method further comprises fetching~~ the processor is operable to fetch the ~~first~~ associated instruction again from the memory to re-execute the ~~first~~ associated instruction.

51 (currently amended). The method of Claim [[23]] 76 wherein the ~~first~~ associated instruction is suspended after being decoded, and the ~~first~~ associated instruction is decoded again when the ~~first~~ associated instruction is being re-executed.

52-53 (canceled).

54 (currently amended). The method of Claim [[23]] 77 wherein the ~~FIFO~~ first resource is a request FIFO for storing requests to process network data, and for at least one first indication the ~~first~~ associated instruction is an instruction to read the request FIFO.

55 (currently amended). The method of Claim [[23]] 77 wherein the ~~FIFO~~ first resource is a command FIFO for storing commands for processing of network data, and for at least one first indication the ~~first~~ associated instruction is an instruction to write one or more commands to the command FIFO.

56 (currently amended). The method of Claim [[23]] 77 wherein the ~~FIFO~~ first resource is a status FIFO for storing status information on reception of network data over a network, and for at least one first indication the ~~first~~ associated instruction is an instruction to read the status FIFO.

57 (canceled).

58 (currently amended). The method of Claim [[57]] 76 wherein suspension of the ~~first~~ associated task and scheduling of ~~each of the one or more other tasks instead~~ another task in lieu of the ~~first~~ associated task does not involve instruction execution.

59-60 (canceled).

61 (currently amended). The method of Claim [[25]] 76 wherein on obtaining of any one of the first and second indications the ~~first~~ associated instruction is operable to be suspended while being executed in an instruction execution pipeline after the pipeline starts execution of one or more other instructions for one or more tasks other than the ~~first~~ associated task but before the pipeline starts execution of any instruction following the ~~first~~ associated instruction for the ~~first~~ associated task.

62 (currently amended). The method of Claim 61 wherein the pipeline comprises an instruction decode stage, and the ~~first~~ associated instruction is suspended after being processed by the decode stage.

63 (currently amended). The method of Claim 61 wherein the ~~first~~ associated instruction is suspended in the pipeline's read stage in which instruction operands are read from storage and presented to instruction execution logic.

64 (currently amended). The method of Claim [[25]] 78 wherein the resource is a FIFO [[is]] shared by the ~~first~~ associated task and at least one other task.

65-68 (canceled).

69 (currently amended). The processor of Claim [[14]] 77 wherein when the ~~first instruction~~ associated task becomes suspended on obtaining the first indication, the ~~first~~ associated instruction is canceled and the ~~first~~ associated instruction is re-executed when the ~~FIFO~~ resource becomes available.

70 (currently amended). The method of Claim [[23]] 78 further comprising, on obtaining the first indication, executing the ~~first~~ associated instruction to completion when the resource ~~FIFO~~ becomes available.

71 (new). A multi-tasking processor for executing computer instructions for a plurality of tasks, the processor being operable to either suspend a task or to re-execute a task's instruction without suspending the task, the processor comprising circuitry for:

scheduling tasks for execution, and executing tasks scheduled for execution;

obtaining first and second indications each of which is associated with a task being executed by the processor and with an instruction being executed for the associated task, each indication being an indication that the associated instruction is not to be executed to completion;

in response to each first indication, (i) suspending the associated instruction without executing the associated instruction to completion, and (ii) suspending the associated task, wherein suspended tasks are not scheduled for execution, the processor being operable to schedule another task for execution in lieu of the associated task; and

in response to each second indication, suspending the associated instruction without executing the associated instruction to completion and without suspending the associated task, and re-executing the associated instruction without suspending the associated task.

72 (new). The processor of Claim 71 wherein at least one first indication is obtained when the associated instruction is accessing a first resource and the first resource is unavailable to the associated instruction, and at least one second indication is obtained when the associated instruction is accessing a second resource and the second resource is unavailable to the associated instruction.

73 (new). The processor of Claim 71 wherein at least one first indication is obtained when the associated instruction is to access an unavailable resource, and the associated task remains suspended until the resource becomes available.

74 (new). The processor of Claim 71 wherein for each first indication, the circuitry is operable to re-execute the associated instruction when the associated task becomes unsuspended and scheduled again for execution.

75 (new). The processor of Claim 71 wherein for each first indication, the circuitry is operable to execute the associated instruction to completion when the associated task becomes unsuspended and scheduled again for execution.

76 (new). A method for executing computer instructions by a multi-tasking processor operable to either suspend a task or to re-execute a task's instruction without suspending the task, the method comprising:

scheduling tasks for execution, and executing tasks scheduled for execution;

obtaining first and second indications each of which is associated with a task being executed by the processor and with an instruction being executed for the associated task, each indication being an indication that the associated instruction is not to be executed to completion;

in response to each first indication, (i) suspending the associated instruction without executing the associated instruction to completion, and (ii) suspending the associated task, wherein suspended tasks are not scheduled for execution, the processor being operable to schedule another task for execution in lieu of the associated task; and

in response to each second indication, suspending the associated instruction without executing the associated instruction to completion and without suspending the associated task, and re-executing the associated instruction without suspending the associated task.

77 (new). The method of Claim 76 wherein at least one first indication is obtained when the associated instruction is accessing a first resource and the first resource is unavailable to the associated instruction, and at least one second indication is obtained when the associated instruction is accessing a second resource and the second resource is unavailable to the associated instruction.

78 (new). The method of Claim 76 wherein at least one first indication is obtained when the associated instruction is to access an unavailable resource, and the associated task remains suspended until the resource becomes available.

79 (new). The method of Claim 76 further comprising, for at least one first indication re-executing the associated instruction when the associated task becomes unsuspended and scheduled again for execution.

80 (new). The method of Claim 76 wherein for each first indication, the processor is operable to execute the associated instruction to completion when the associated task becomes unsuspended and scheduled again for execution.